PATENT

#10
LDD
9-5-02

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Paul E. McKenney Art Unit: 2151

Application No.: 09/127,085

Filed: July 31, 1998

For: HIGH SPEED METHOD FOR MAINTAINING A SUMMARY OF THREAD ACTIVITY FOR MULTIPROCESSOR COMPUTER SYSTEMS

Examiner: Majid A. Banankhah

Date: January 22, 2002

CERTIFICATE OF MAILING

I hereby certify that this paper and the documents referred to as being attached or enclosed herewith are being deposited with the United States Postal Service on January 22, 2002 as First Class Mail in an envelope addressed to: BOX AF, ASSISTANT COMMISSIONER FOR PATENTS, WASHINGTON, D.C. 20231.

Robert F. Scotti, Attorney for Applicant

APPEAL BRIEF

BOX AF
ASSISTANT COMMISSIONER FOR PATENTS
Washington, DC 20231

Sir:

This brief is in furtherance of the Notice of Appeal filed December 3, 2001. The fee required under 37 CFR 1.17(b) is enclosed.

CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is IBM Corporation through acquisition of the original assignee of record, Sequent Computer Systems, Inc.

II. RELATED APPEALS AND INTERFERENCES

None.

III. STATUS OF CLAIMS

An amendment after the final rejection was filed on September 4, 2001.

An Advisory Action (paper no. 7) dated November 21, 2001, states that the amendment after final dated September 4, 2001, was not entered.

Claims 1-10 are finally rejected and appealed. New claims 7-10 were not entered after final, but are dependent claims and are believed to place the application in better form for consideration on appeal. Other amendments made in the amendment after final also place the case in better form for consideration on appeal and were improperly not entered, as further described below.

IV. STATUS OF AMENDMENTS

An amendment after final remains unentered.

V. SUMMARY OF THE INVENTION

A high-speed method for maintaining a summary of thread activity reduces the number of remote-memory operations for an n processor, multiple node computer system from n^2 to (2n-1) operations. The method uses a hierarchical summary-of-thread-activity data structure that includes structures such as first and second level bit masks. (Specification, page 27, lines 7-21). The first level bit mask 302 (see Figure 8) is accessible to all nodes and contains a bit per node, the bit indicating whether the corresponding node contains a processor that has not yet passed through a quiescent state. The second level bit mask 304, 306, 308 (Figure 8) is local to each node and contains a bit per processor per node, the bit indicating whether the corresponding processor has not yet passed through a quiescent state. The method includes determining from a data structure on the processor's node (such as a second level bitmask) if the processor has passed through a quiescent state. (Page 27, lines 22-29). If so, it is then determined from the data structure if all other

processors on its node have passed through a quiescent state. (Page 27, lines 22-29). If so, it is then indicated in a data structure accessible to all nodes (such as the first level bitmask) that all processors on the processor's node have passed through a quiescent state. (Page 27, lines 22-29). The local generation number can also be stored in the data structure accessible to all nodes. If a processor determines from this data structure that the processor is the last processor to pass through a quiescent state, the processor updates the data structure for storing a number of the current generation stored in the memory of each node.

VI. ISSUES

(A) Whether the claims fail to comply with 35 U.S.C. § 112, second paragraph, as omitting essential elements.

(B) Whether the difference between the claimed invention and the cited publications are such that the claimed invention would have been obvious under 35 U.S.C. § 103 at the time the invention was made to a person having ordinary skill in the art.

VII. GROUPING OF CLAIMS

For reasons detailed below, each of the independent claims 1, 2, and 3 is independently patentable. The dependent claims each contain limitations that further distinguish over the art of record. However, since the limitations of the independent claims are sufficient to distinguish this art and to facilitate the Board's consideration of this appeal, Applicants group the claims for purposes of this appeal as follows:

The patentability of claim 1 stands alone.

The patentability of claim 2 stands alone.

The patentability of claims 4-10 stand or fall with the patentability of claim 3.

VIII. ARGUMENT

A. Rejection of Claims Under 35 U.S.C. § 112, second paragraph

Claim 2

Applicants respectfully request reversal of the Examiner's rejection of claim 2.

First, it should be noted that the Amendment After Final dated September 4, 2001, should have been entered because it clearly places the case in better condition for appeal. For example, the

Examiner rejected claim 2, in part, because "[i]t is not clear whether 'physical memory' in line 14 is the same as 'memory' in line 17." (Final Office action page 2). The Amendment after final changed line 17 to say "physical" memory, thus correcting the Examiner's concern. Clearly, other amendments were also made to place the application in better condition for appeal.

Additionally, the Examiner argued that there is no relationship between the "physical memory" and the "data structure" and any means of the claims. (Final Office action, page 2). In response, Applicants' amended claim 2 to include such a relationship. For example, claim 2 was amended to state that the "data structure [is] stored in the physical memory on each node." Thus, every element in the claim ties together the data structure and the physical memory. Additionally, claim 2 was amended to make clear that the data structure is stored in memory on each node. This amendment was in response to the Examiner asking "is this data structure unique for all the nodes or every node has its own data structure." (Final Office action, page 2).

Claim 3

Claim 3 was also amended to clarify the claim as requested by the Examiner. (Final Office action, page 3). The Examiner refused to enter the amendment after final because the addition of language to claim 3 regarding the first, second, and third data structures (see advisory action dated 11/21/02). However, applicants were amending claim 3 specifically to overcome the 112 rejection by the Examiner. In the Final Office action at page 3, the Examiner indicated that it was unclear whether a recited data structure was the same data structure as a previous data structure or a different one ("Is it the same data structure recited online 25", Office action at page 3). Applicants simply amended claim 3 to clarify that they may be different data structures. Consequently, the amendment should have been entered.

B. Rejection of Claims Under 35 U.S.C. § 103

The Examiner rejected claims 1-6 under 35 U.S.C. § 103 as unpatentable over Slingwine (U.S. Patent No. 5,727,209) in view of Roche et al. (U.S. Patent No. 4,916,697).

Claim 1

Applicants respectfully request reversal of the Examiner's rejection of claim 1.

The application describes at pages 27-28 and Figure 8 the following:

> Fig. 8 shows the structure of a hierarchical bit mask constructed in accordance with the invention. A summary-of-thread- activity-data structure 300 includes a first level bit mask 302 stored in memory and containing a bit per node such as Q0 representing a first node, Q1 representing a second node, and Qn representing an nth node of the compute system. The bit indicates whether the corresponding node contains a processor that has not yet passed through a quiescent state. The data structure 300 also includes a second level bit mask stored in the memory of each processing node such as bit mask 304, bit mask 306, and bit mask 308. Each second level bit mask contains a bit per processor such as, for bit mask 304, C0 representing a first processor on node Q0, C1 representing a second processor on the node, and C3 representing a fourth processor on the node. The bit indicates whether the corresponding processor has not yet passed through a quiescent state. This hierarchical data structure 300 allows processors to avoid many of the expensive remote references that they would otherwise have to make to a global bit mask.
>
> A method for maintaining a summary of thread activity in accordance with the invention includes determining from a data structure on the processor's node (such as a second level bitmask 304) if the processor has passed through a quiescent state. If so, it is then determined from the data structure 304 if all other processors on its node have passed through a quiescent state. If so, it is then indicated in a data structure accessible to all nodes (such as the first level bitmask 302) that all processors on the processor's node have passed through a quiescent state.
>
> If the processor determines from a data structure on the processor's node such as 304 that the processor has not passed through a quiescent state, a callback processor is activated. The callback processor checks in the manner described above if the processor has passed through a quiescent state and, if so, has the processor then indicate in the data structure that it has passed through a quiescent state.

Claim 1 calls for a first level bitmask and a second level bitmask. The first level bitmask contains "a bit per node, the bit indicating whether the corresponding node contains a processor that has not yet passed through a quiescent state."

The second level bit mask contains "a bit per processor associated with a particular node...the bit indicating whether the corresponding processor has not yet passed through a quiescent state."

The Examiner seems to focus on the preamble of the claim and argues that Slingwine teaches "states of threads" and a "summary of thread activity." (Final Office action, page 3). However, the body of the claim calls for the first level and second level bitmasks to indicate whether the processor has not yet passed through a quiescent state. For this part of the claim, the Examiner argues it is well known at the time of the invention to use level bit masks for an indication of passing through a quiescent state. (Final Office action, page 4, citing Roche at Col. 8, lines 5-30). However, the portion of Roche the Examiner cites has nothing to do with indicating whether a node contains a processor that has not yet passed through a quiescent state. Nor does Roche teach a bit per processor associated with a particular node, the bit indicating whether the corresponding processor has not yet passed through a quiescent state.

Rather, Roche specifically states that "the associated bits indicating into which levels the particular error is to be classified." (Roche, 8:5-8). The remaining portion of Roche from Col. 8, lines 5-30, clearly indicates that Roche is discussing error detection, not whether or not a processor has passed through a quiescent state.

Thus, the Examiner's decision to reject claim 1 should be reversed.

Claim 2

Applicants respectfully request reversal of the Examiner's rejection of claim 2.

The application describes FIG. 9 as follows:

> Fig. 9 shows how a copy of the global rcc_curgen variable 310 has been replicated in the per-node pq_rcc_curgen variables such as 312, 314, and 316. These variables are updated in lockstep to avoid expensive remote-memory references. In contrast, the per-processor rclockgen variables 318, 320, 322 indicate on which read-copy generation that the corresponding processor's rclock curlist is waiting. The rclockgen variables are not updated in lockstep and need not have the same value.
>
> These variables form a data structure for storing a number of the current generation of data elements being processed by a processor on a node, the data structure comprising a variable containing the current generation number stored in the memory of each node.
>
> If a processor determines from data structure 302 (which is accessible to all nodes) that the processor is the last processor to pass through a quiescent state, the processor updates a data structure such as 318, 320,

> or 322 for storing a number of the current generation stored in the memory of each node.
>
> Also, if a processor determines from the data structure 302 that it is the last processor to pass through a quiescent state, a callback processor is activated to determine if there are callbacks waiting for a subsequent generation, and, if so, updates a data structure on each node such as structures 304, 306, and 308) and the data structure 302 to indicate that all processors need to pass through an additional quiescent state.

Claim 2 requires "a data structure stored in the physical memory on each node for storing a number of the current generation of data elements being processed by a processor on a node."

The Examiner makes almost no mention of any specific limitations of claim 2. In fact, Applicants specifically pointed out in an amendment dated March 19, 2001, that "the Examiner did not demonstrate any reasoning for finding claims 2 or 3 obvious..." (Response of March 19, 2001, at page 5).

Nonetheless, the Examiner continues to provide limited reasoning for rejecting claim 2. Claim 2 requires a data structure stored in the physical memory on each node, which Slingwine does not disclose. Additionally, claim 2 has been amended to further remove it from Slingwine by requiring that "the current generation number on the nodes are updated in lockstep so the nodes have local access...". Slingwine does not store the current generation on each node and thus would have no reason to update the current generation number in lockstep on the nodes.

Thus, the Examiner's decision to reject claim 2 should be reversed.

Claim 3

Applicants respectfully request reversal of the Examiner's rejection of claim 3.

Claim 3 has numerous limitations that the Examiner does not appear to address. For example, claim 3 requires "determining from a first data structure on the processor's node if the processor has passed through a quiescent state." The Examiner does not appear to address this language in the rejection.

Nor does the Examiner mention or point out where in the references the following language of claim 3 is disclosed: "if so, determining from a second data structure on the processor's node if all other processors on its node have passed through a quiescent state."

Nor does the Examiner mention or point out where in the references the following language of claim 3 is disclosed: "if so, indicating in a third data structure accessible to all nodes that all processors on the processor's node have passed through a quiescent state."

Applicants believe it is the Examiner's burden to show why the invention is not patentable. Applicants requested the Examiner to more specifically identify where the above limitations are found in the references, but have received essentially the same rejection again with no particulars identified.

Applicants would also like to take issue with the Examiner's use of Roche. The Examiner states "it would have been obvious for one ordinary skill in the art at the time of the invention was made to use 'level bit mask registry as an indication of a state of a processor' of Roche." (Final office action, page 4 (emphasis added)). The Examiner appears to be quoting Roche, but Roche does not seem to have this quoted language. For example, Applicant's representative searched Roche for the words "mask registry" and found nothing. This quoted language also does not appear in the claims at issue. Applicants representative is unsure what the Examiner is quoting, but believes it is not an accurate representation to place a quote as such in reference to Roche.

Thus, the Examiner's decision to reject claim 3 should be reversed.

Claims 4-10

Claims 4-10 depend from claim 3 and should be allowable for the reasons stated above.

IX. CONCLUSION

Accordingly, the rejection of claims 1-10 should be reversed and all claims passed to issue.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

By _____
Robert F. Scotti
Registration No. 39,830

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 226-7391
Facsimile: (503) 228-9446
cc: Client

APPENDIX A
CLAIMS ON APPEAL

1. (Amended) In a multiprocessor computer system having multiple interconnected processing nodes each with one or more processors and physical memory, a data structure for storing execution history data indicative of states of threads that are used for providing mutual exclusion between current and next generation data elements, comprising:

a first level bit mask stored in physical memory accessible to all nodes and containing a bit per node, the bit indicating whether the corresponding node contains a processor that has not yet passed through a quiescent state; and

a second level bit mask stored in the physical memory of each processing node and containing a bit per processor associated with a particular node identified in the first level bit mask, the bit indicating whether the corresponding processor has not yet passed through a quiescent state.

2. (Amended) In a multiprocessor computer system having multiple interconnected processing nodes each with one or more processors and physical memory, a data structure stored in the physical memory on each node for storing a number of the current generation of data elements being processed by a processor on a node, the data structure comprising a variable stored in the physical memory of each node and containing the current generation number, wherein the current generation number on the nodes are updated in lockstep so that the nodes have local access to copies of the current generation number.

3. (Amended) In a multiprocessor computer system having multiple interconnected processing nodes each with one or more processors and physical memory, a method for a processor to maintain a summary of thread activity as part of a method for providing mutual exclusion between current and next generation data elements, comprising:

determining from a first data structure on the processor's node if the processor has passed through a quiescent state;

if so, determining from a second data structure on the processor's node if all other processors on its node have passed through a quiescent state; and

if so, indicating in a third data structure accessible to all nodes that all processors on the processor's node have passed through a quiescent state.

4. (Amended) The method of claim 3 wherein if the processor determines from the first data structure on the processor's node that the processor has not passed through a quiescent state, having a callback processor check if the processor has passed through a quiescent state and, if so, having the processor indicate in the first data structure that it has passed through a quiescent state.

5. (Amended) The method of claim 3 wherein if the processor determines from the third data structure accessible to all nodes that the processor is the last processor to pass through a quiescent state, having the processor update a fourth data structure stored in the memory of each node for storing a number of the current generation of elements being processed on the node.

6. (Amended) The method of claim 3 wherein if the processor determines from the third data structure accessible to all nodes that it is the last processor to pass through a quiescent state, having a callback processor determine if there are callbacks waiting for a subsequent generation, and, if so, updating the first data structure on each node and the third data structure accessible to all nodes to indicate that all processors need to pass through an additional quiescent state.

7. (New) The method of claim 3, wherein the first, second, and third data structures are the same data structure.

8. (New) The method of claim 3, wherein the first and second data structures are the same data structure.

9. (New) The method of claim 3, wherein the second and third data structures are the same data structure.

10. (New) The method of claim 3, wherein the first and third data structures are the same data structure.